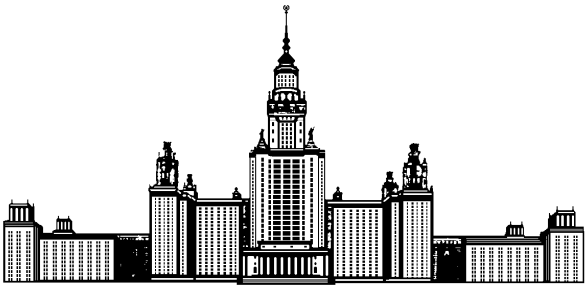


Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Липкина Анна Львовна

Распознавание текста по структуре скелета букв

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
д. т. н., профессор
Л. М. Местецкий

Москва, 2018

Содержание

Аннотация

В настоящей работе предложена математическая модель понятия графемы и алгоритм её построения на основе непрерывного морфологического подхода. Такой подход подразумевает, что основой для модели является скелетный граф бинарного изображения.

Также в работе предложен метод распознавания изображения печатного текста в кириллических шрифтах на основе математической модели графемы. Алгоритм распознавания основывается на выделении признаков из математической модели графемы и последующем построении классификатора графем.

Проведенные эксперименты показывают, что предлагаемая математическая модель является работоспособной: алгоритм классификации, построенный на выделенных из неё признаках, показывает высокое качество распознавания текста в различных вариантах шрифта и в различных качествах изображения текста.

1 Введение

Когда грамотный человек читает текст, он по виду символа может сразу определить, какую букву этот символ изображает. Причём может это сделать вне зависимости от различных вариантов художественного начертания символа (с засечками, наклонного, прямого, декоративного и т. д. [1]). То есть существует некий «образ» буквы, который легко узнаваем человеком и легко отличим от «образов» других букв. Такой «образ» называется *графемой* [2].

Более формально соотношение между буквами и графемами можно дать через определения следующим образом:

Определение 1.1. *Буква* — отдельный символ алфавита.

Часто буква соответствует отдельному звуку в устной речи (например, в русском алфавите 33 буквы).

В процессе развития письменности и скорописи [3][4] у букв появилось несколько начертаний: строчное и прописное, а позднее — различные варианты написания одной и той же буквы. Зачастую эти написания могут довольно сильно различаться, хотя и обозначают они произношение одного и того же звука, например: **A** и **a**. Для описания таких различий вводится понятие графемы:

Определение 1.2. *Графема* — единица письменности.

Другими словами, графема — некоторый графический примитив, имеющий вид геометрического графа и изображающий канонические начертания буквы. Графему можно представлять себе как изображения букв в тонком шрифте, например, как на рис. 1.

Рис. 1: Изображения букв в шрифте Lato и выделенные вершины геометрических графов

Графемы должны обладать следующими свойствами:

1. Любые две графемы хорошо отличимы между собой.

2. Пусть изображения I_1 и I_2 представляют одну и ту же графему. Тогда различие между I_1 и I_2 *несущественное*. Таким образом, похожесть определяется некоторой *мерой сходства* между I_1 и I_2.

Стоит отметить, что буквы и графемы не эквивалентны друг другу. Каждая буква может описываться несколькими графемами, например, как на рис. 2.

Рис. 2: Различные варианты графем строчной буквы «т»

Что такое *несущественное* различие, конечно, понятно на интуитивном уровне: например, на рис. 3 изображены различные варианты графемы прописной буквы «А», но грамотный человек, видя их, понимает, что это изображение одной и той же буквы. В данной работе, в частности, будет формально описано, что такое *несущественное* различие, то есть описана *мера сходства* между графемами.

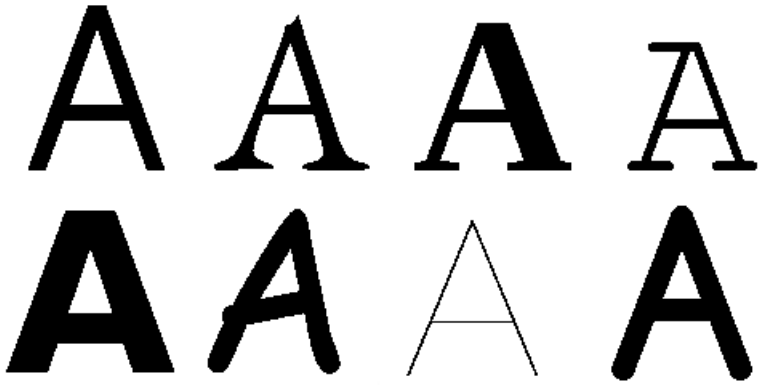


Рис. 3: Различные варианты графемы прописной буквы «А»

Существует много различных вариантов графем, отличающихся начертанием: жирные, курсивные, с засечками и т. д. [5] [1]. Однако интуитивно понятно, что такие начертания обладают некой «общей» конструкцией.

Цель данной работы — составить математическое описание такой «общей» конструкции и проверить гипотезу о том, что такого описания достаточно для распознавания букв в большинстве шрифтов.

Стоит также отметить, что на сегодняшний день задача автоматического преобразования печатных текстов в цифровой вид, иначе именуемого *оптическим распознаванием символов* (optical character recognition, OCR), остается актуальной, несмотря на большое число существующих OCR-систем, поскольку востребованы недорогие компактные алгоритмы для частных приложений в типографике (распознавание шрифтов по изображениям текста), в оцифровке документов (таблицы, чеки, накладные) и т. п. Существует множество подходов к решению данной задачи: например, структурный анализ изображения [6] или применение нейросетевых классификаторов [7].

2 Постановка задачи

Задача фактически состоит из двух частей:

1. построение математической модели графемы;
2. разработка алгоритма на основе построенной модели, распознающего букву по математической модели графемы, и, как следствие, извлекающего текст из цифрового изображения текста (см. рис. 4).

Рис. 4: Пример бинаризованного входного изображения текста

Идея построения математической модели графемы состоит в построении скелетного графа бинарного изображения буквы и удаления некоторых рёбер из него, то ест поиск некоторого подграфа, отражающего свойства букв.

Концептуальный подход к выделению (алгоритму распознавания) графем состоит в следующем: строится скелет бинарного изображения буквы, в этом графе неким образом ищется подграф, эквивалентный по мере сходства эталонному математическому описанию графемы. Поиск подграфа происходит путём «стрижки» скелета. Алгоритм стрижки скелета основан на следующей идее: удаление по одному рёбер скелетного графа и последующее сравнение с эталонным графом. Стрижка происходит до тех пор, пока текущий граф не будет похож на эталонный по некоторой мере сходства, которая основывается на выделении признаков из скелетного графа.

Прежде чем приступить к описаниям модели и алгоритма, определим несколько базовых понятий.

Определение 2.1. *Фигура* — множество точек на плоскости.

Определение 2.2. *Контурное представление фигуры* — аппроксимация границ фигуры многоугольником минимального периметра.

До аппроксимации граница фигуры представляет собой многоугольник с прямыми углами, стороны которого параллельны осям координат (см. рис. 5)

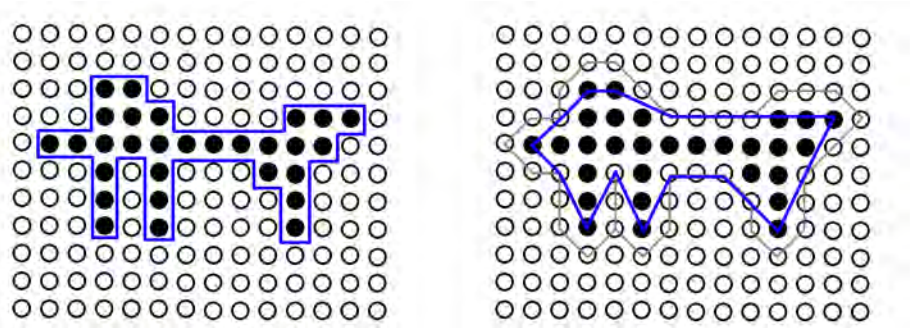


Рис. 5: Граница фигуры до аппроксимации (слева) и после (справа)

Далее подразумевается, что граница фигуры аппроксимирована многоугольником минимального периметра.

Определение 2.3. *Пустой круг фигуры* — круг, целиком лежащий в фигуре.

Определение 2.4. *Вписанный пустой круг фигуры* — пустой круг фигуры, не содержащийся ни в каком другом пустом круге фигуры.

Определение 2.5. *Скелетное представление фигуры* — множество центров всех вписанных пустых кругов фигуры (см. рис. 6).

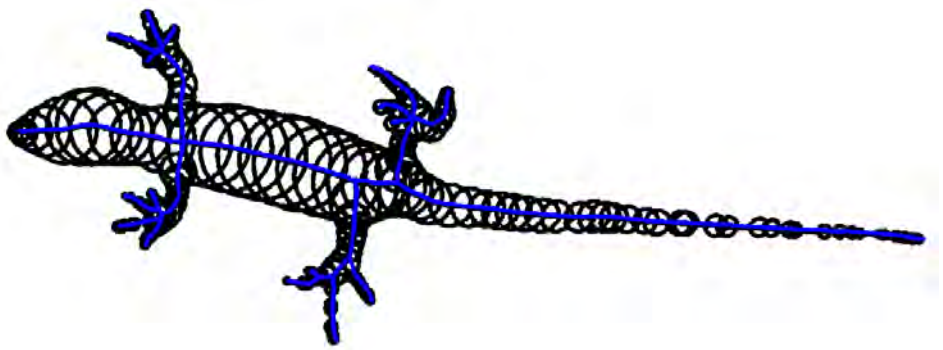


Рис. 6: Скелетное представление фигуры

В действительности скелетное представление фигуры представляет из себя граф $\mathcal{S}$, называемый *скелетом (скелетным графом)* фигуры (см. рис. 7). Вершинами графа являются центры вписанных пустых кругов, имеющих либо одну, либо три общих точки с границей фигуры, а рёбрами — линии из центров вписанных пустых кругов, касающихся границы ровно в 2 точках. Более подробно скелетное представление фигуры рассматривается в [8].

Рис. 7: Пример фигуры (слева) и её скелетного графа (справа)

Определение 2.6. *Силуэт скелетного графа* — фигура, состоящая из объединения всех вписанных пустых кругов, центры которых лежат в скелетном графе $\mathcal{S}$. Обозначение: $\mathcal{V}_{\mathcal{S}}$.

Определение 2.7. *Стрижка скелетного графа (с параметром α)* — процесс регуляризации скелетного графа $\mathcal{S}$, основанный на удалении несущественных рёбер из скелетного графа (см. рис. 8). В процессе такого удаления возникает минимальный подграф $\mathcal{S}'$ исходного скелетного графа, для которого выполняется $H(\mathcal{V}_{\mathcal{S}}, \mathcal{V}_{\mathcal{S}'}) \leqslant \alpha$, где $H(\mathcal{V}_{\mathcal{S}}, \mathcal{V}_{\mathcal{S}'})$ — хаусдорфово расстояние [9] между силуэтом скелетного графа $\mathcal{S}$ и силуэтом скелетного графа $\mathcal{S}'$.

Рис. 8: Пример скелета без стрижки (слева) и со стрижкой (справа)

3 Метод решения

Как было замечено ранее, решение поставленной задачи состоит из двух этапов: построения математической модели графем, и последующего распознавания текста на основе построенных моделей.

3.1 Построение математической модели графемы

Для построения математической модели графемы предлагается сделать два шага:

1. Сегментация изображения текста на изображения отдельных символов (графем).
2. Выделение структурного описания (математической модели) изображения каждой графемы.

Для упрощения в данной работе считается, что решение первого шага не составляет труда: на этапе обучения приходят уже выделенные изображения отдельных символов, а на этапе тестирования — изображение такого качества, что соседние символы разделимы между собой (т. е. нет «слипания» символов).

Второй шаг разделяется на следующие действия:

1. Получение скелетного графа из изображения буквы.
2. Агрегирование скелетного графа.
3. Обработка скелетного графа, а именно удаление шумовых рёбер.

3.1.1 Получение скелетного графа из изображения буквы

Построение скелетного графа фигуры подробно описывается в [8] и содержит следующие основные шаги:

1. Аппроксимация исходной фигуры F многоугольником минимального периметра M.
2. Построение диаграммы Вороного [8] для вершин и сторон многоугольника M.
3. Удаление некоторых отрезков из диаграммы Вороного.
4. Аппроксимация параболических рёбер диаграммы Вороного прямолинейными рёбрами.

После построения скелетного графа производится его последующая стрижка с параметром α. Она производится для того, чтобы выделить основные элементы скелетного графа, не зависящие от незначительных изменений границ изображения символа.

3.1.2 Агрегирование скелетного графа

Полученный скелетный граф содержит только следующие типы вершин:

- вершины степени 1 (листья);
- вершины степени 2;
- вершины степени 3 (развилки).

Основную информацию о скелетном графе несут листья и развилки, а также виды связей между ними. Чтобы выделить эти связи, производится операция *агрегирования скелетного графа:* «склеивание» в одну цепь всех таких последовательных рёбер, инцидентные вершины которых имеют степень либо 1, либо 2. После такой «склейки» в качестве вершин остаются только листья и развилки (см. рис. 9).

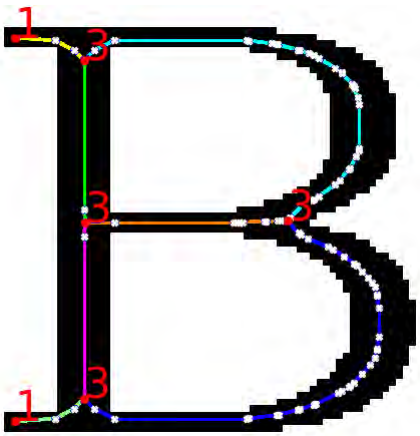


Рис. 9: Пример агрегированного скелетного графа. Белым цветом отмечены вершины степени 2 исходного графа.

Ниже приведён алгоритм агрегации скелетного графа:

Алгоритм 1 Построение агрегированного скелета

пока все рёбра не обработаны:
  Начать новое рёбро (цепь) R агрегированного скелета
  Выбрать необработанное ребро E скелета
  Добавить E к R
  Пометить E как обработанное
  пока можно наращивать текущую цепь
    Рассмотреть инцидентные с E рёбра по обоим концам
    если конец ребра E степени 1 или 3 **то**
      Дальше в этом направлении рёбра не наращивать
    если конец ребра E степени 2 и инцидентное ребро I_E ещё не было обработано **то**
      Добавить I_E к R
      Пометить I_E как обработанное
      Продолжить наращивание, начиная с I_E
    иначе
      Дальше в этом направлении рёбра не наращивать

После применения алгоритма агрегации скелетный граф представляет собой гиперграф $\mathcal{S}_{agg,1}$, вершинами которого являются листы и развилки, а рёбрами — выделенные в процессе агрегирования цепочки.

Обозначения и понятия

Для дальнейшего описания алгоритма понадобятся следующие обозначения и понятия:

1. Рассматривается входное бинарное изображение символа. $\mathcal{B}$ — минимальная по площади прямоугольная рамка с горизонтальными и вертикальными сторонами, ограничивающая данный символ. $\mathcal{B}_H$ и $\mathcal{B}_W$ — высота и ширина рамки $\mathcal{B}$ соответственно.

2. Пусть e — ребро неагрегированного скелетного графа $\mathcal{S}$. $v_1(e), v_2(e)$ — концевые вершины этого ребра без учёта какого-либо порядка.

3. $l(e)$ — длина ребра e. Она вычисляется через евклидово расстояние между двумя точками $v_1(e)$ и $v_2(e)$:

$$l(e) = \sqrt{\big(v_1(e)_x - v_2(e)_x\big)^2 + \big(v_1(e)_y - v_2(e)_y\big)^2}.$$

4. Аналогично для ребра (цепочки) e_{agg} гиперграфа $\mathcal{S}_{agg}$ через $v_1(e), v_2(e)$ обозначаются концевые вершины этой цепочки;

5. Ребро e_{agg} обозначенного гиперграфа $\mathcal{S}_{agg}$ состоит из n последовательных рёбер исходного графа $\mathcal{S}$, попавших в данную цепочку e_{agg}: $\{e^1_{agg}, e^2_{agg}, \ldots, e^n_{agg}\}$.

6. $l(e_{agg})$ — длина цепочки e_{agg}. Она считается как сумма длин всех входящих в данную цепочку рёбер:

$$l(e_{agg}) = \sum_{i=1}^{n} l(e^i_{agg}).$$

7. $\deg v$ — степень вершины v.

Также вводятся понятия *аппроксимирующей дуги ребра* e_{agg}, *центрального угла дуги* и *угла кривизны цепочки.*

Определение 3.1. Пусть дан отрезок $d = [v_1(e_{agg}), v_2(e_{agg})]$. Среди всех вершин цепочки e_{agg} находится вершина v_h, наиболее удалённая от отрезка d. По трём точкам $v_1(e_{agg}), v_2(e_{agg}), v_h$ строится окружность. Тогда *аппроксимирующей дугой* называется дуга наименьшей длины, ограниченная точками $v_1(e_{agg}), v_2(e_{agg})$ (см. рис. 10).

Определение 3.2. Пусть ω — окружность, L — некоторая дуга окружности, концами которой являются точки A и B. Пусть O — центр окружности ω. Тогда $\angle BOA$ — *центральный угол* дуги L (см. рис. 10).

Определение 3.3. *Угол кривизны цепочки* — центральный угол её аппроксимирующей дуги.

Замечание. В случае, когда три точки $v_1(e_{agg}), v_2(e_{agg}), v_h$ лежат на одной прямой или когда в цепочке e_{agg} нет вершин, угол кривизны цепочки полагается равным 0.

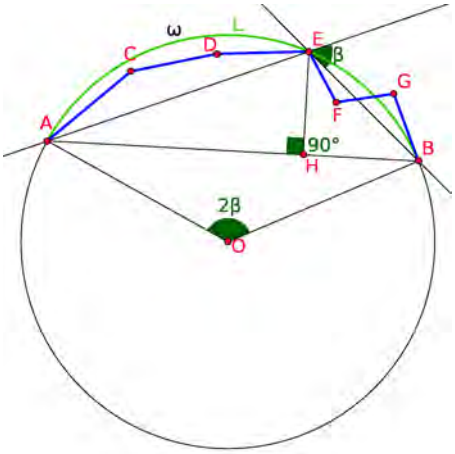
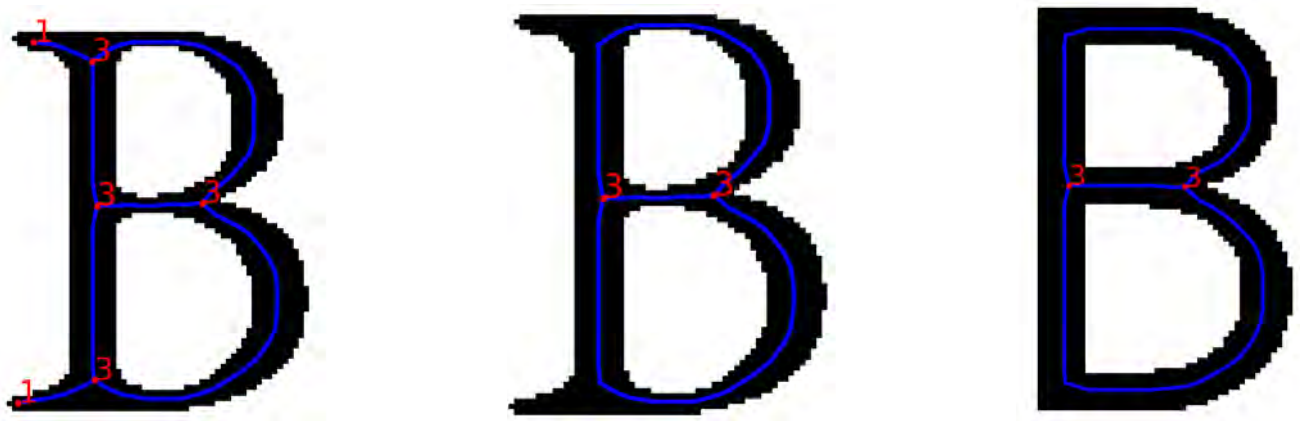


Рис. 10: Пример аппроксимирующей цепочку $[A, C, D, E, F, G, B]$ дуги L и центрального угла BOA

3.1.3 Удаление шумовых рёбер

После проделанной регуляризации скелетного графа и его последующей агрегации в $\mathcal{S}_{agg,1}$ все ещё могут содержаться шумовые рёбра. Это проявляется на буквах, изображённых в шрифтах с засечками [1].

Засечки это некое украшение для буквы, и их наличие или отсутствие не мешает человеку распознать, какая именно буква изображена. Таким образом в модель графемы буквы засечки не должны входить. Поэтому следующий этап построения математической модели графемы — удаление из $\mathcal{S}_{agg,1}$ рёбер, являющихся засечками (см. рис. 11). Множество рёбер гиперграфа $\mathcal{S}_{agg,1}$, являющихся засечками, обозначим через $\mathcal{E}_{\mathcal{S}}$.



(a) **(b)** **(c)**

Рис. 11: 11a: скелет буквы в шрифте с засечками; 11b: тот же скелет с удалёнными рёбрами из $\mathcal{E}_{\mathcal{S}}$; 11c: скелет буквы в шрифте без засечек.

Для множества $\mathcal{E}_{\mathcal{S}}$ можно выделить следующие признаки:

- $|\mathcal{E}_{\mathcal{S}}| \geqslant 2$, то есть если засечки в скелетном графе присутствуют, то их не меньше двух.
- $\forall\ e_{agg} \in \mathcal{E}_{\mathcal{S}}$ характерны следующие признаки:
  - ровно одна из вершин $\{v_1(e_{agg}), v_2(e_{agg})\}$ — листовая, и ровно одна из них является развилкой;
  - длина ребра $l(e_{agg})$ не превышает некоторого порога $\mathcal{L}(\mathcal{B})$;
  - центральный угол 2β аппроксимирующей e_{agg} дуги не меньше некоторого порога $\mathcal{A}$.

Таким образом, алгоритм удаления шумовых рёбер из $\mathcal{S}_{agg,1}$ состоит из следующих шагов:

1. Определение множества $\mathcal{E}_{\mathcal{S}}$ на основе его признаков.
2. Удаление из агрегированного скелетного графа $\mathcal{S}_{agg,1}$ всех рёбер из $\mathcal{E}_{\mathcal{S}}$.

Так как после удаления могут возникнуть вершины степени 2, необходимо провести повторное агрегирование скелета. Гиперграф, полученный после удаления рёбер и повторной агрегации, обозначим через $\mathcal{S}_{agg,2}$. Этот гиперграф и является предлагаемой математической моделью графемы.

3.2 Распознавание графем

На этом этапе из $S_{agg,2}$ будут выделяться признаки для последующего построения классификатора графем.

3.2.1 Генерация признаков

Поскольку скелетный граф каждого символа имеет свою структуру, то выделение сразу для всех букв фиксированного числа сложных признаков, описывающих структуру скелетного графа, невозможно. Поэтому в данном методе предлагается выделять 2 типа признаковых описаний: *признаки верхнего уровня* $\mathcal{F}_a$ и *признаки нижнего уровня* $\mathcal{F}_d$. Они обладают следующими свойствами:

- Если из гиперграфов $\mathcal{S}'_{agg,2}, \mathcal{S}''_{agg,2}$ выделяются одинаковые признаки верхнего уровня $\mathcal{F}'_a = \mathcal{F}''_a$, то признаки нижнего уровня $\mathcal{F}'_d$ и $\mathcal{F}''_d$ лежат в одном признаковом пространстве.

- Если из гиперграфов $\mathcal{S}'_{agg,2}, \mathcal{S}''_{agg,2}$ выделяются различные признаки верхнего уровня $\mathcal{F}'_a \neq \mathcal{F}''_a$, то признаки нижнего уровня $\mathcal{F}'_d$ и $\mathcal{F}''_d$ лежат в разных признаковых пространствах.

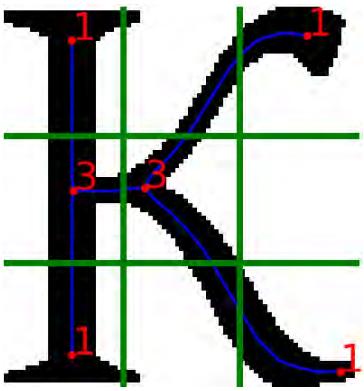
Иными словами, математические модели графем разбиваются на такие классы, что в каждом классе возможно выделить признаки, лежащие в одинаковых признаковых пространствах, а признаковые пространства из двух различных классов не совпадают.

Признаки верхнего уровня

Идея построения признаков верхнего уровня основывается на анализе положения вершин в гиперграфе $\mathcal{S}_{agg,2}$. Рамка $\mathcal{B}$, в которую заключена графема, разбивается на n равных частей горизонтальными прямыми и на m равных частей — вертикальными. Значения n и m должны быть не очень большими в силу возможных вариаций вида графемы в различных шрифтах. В каждом из получившихся $n \cdot m$ прямоугольников считается количество листьев и количество развилок, и эти количества добавляются в признаковое описание верхнего уровня.

Помимо этого в качестве признака верхнего уровня рассматривается количество компонент связности графемы внутри рамки $\mathcal{B}$.

Таким образом, для каждой графемы признаковое описание $\mathcal{F}_a$ состоит из $2nm+1$ элементов (см. рис. 12).



(a)

1	**0**	**1**
0	**0**	**0**
1	**0**	**1**

(b)

0	**0**	**0**
1	**1**	**0**
0	**0**	**0**

(c)

Рис. 12: 12a: скелет $\mathcal{S}_{agg,2}$ буквы «К» и разбиение рамки на 9 прямоугольников ($n = m = 3$); 12b: количества листьев в каждом из прямоугольников; 12c: количества развилок в каждом из прямоугольников.

Признаки нижнего уровня

На этом этапе признаки нижнего уровня генерируются исходя из того, что признаковое описание верхнего уровня $\mathcal{F}_a$ зафиксировано. В свою очередь, это означает, что фактически зафиксировано строение гиперграфа $\mathcal{S}_{agg,2}$: для каждого из $n \cdot m$ прямоугольников разбиения известно количество листьев и развилок, попавших в него, а это также означает, что известно количество рёбер гиперграфа, связанных с каждым прямоугольником разбиения. Таким образом, теперь появляется возможность генерировать фиксированное число признаков для каждого из $n \cdot m$ прямоугольников. Сами прямоугольники для определенности признакого пространства упорядочим слева направо и сверху вниз.

Признаковое описание нижнего уровня $\mathcal{F}_d$ предлагается генерировать из структуры рёбер $\mathcal{S}_{agg,2}$.

Генерация признаков из ребра

Пусть $[A, B]$ — ребро гиперграфа $\mathcal{S}_{agg,2}$. Фиксируется маска разбиения этого ребра на k частей:

$$\mathcal{Z}_k = [z_1, z_2, \ldots, z_k], \quad z_j \in (0, 1) \;\; \forall j = \overline{1, k}.$$

Фиксируется стартовая вершина (не ограничивая общности считаем, что это A). Применяем разбиение $\mathcal{Z}_k$ к ребру $[A, B]$, начиная от вершины A, следующим образом: ребро $[A, B]$ разбивается k точками, считая от точки A, на $k+1$ отрезков s_i так, что:

$$\sum_{i=1}^{j} l(s_i) = z_j l([A, B]) \;\; \forall j = \overline{1, k}.$$

Пусть концы отрезков s_i имеют координаты C_{i-1}, C_i:

$$s_i = [C_{i-1}, C_i] \;\; \forall i = \overline{1, k+1}.$$

Заметим, что $C_0 = A$ и $C_{k+1} = B$. Также отдельно обозначим $\overrightarrow{b} = \overrightarrow{AC_1}$.
Пример разбиения можно увидеть на рис. 13.

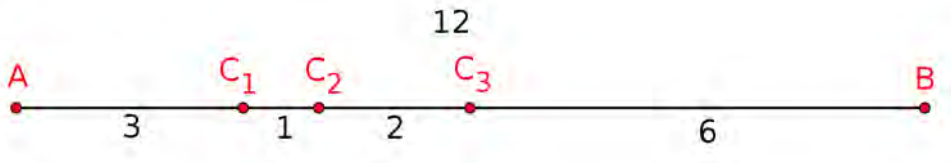


Рис. 13: Пример разбитого маской $\mathcal{Z}_3 = \left[\frac{1}{4}, \frac{1}{3}, \frac{1}{2}\right]$ ребра $[A, B]$ длины 12 со стартовой вершиной A

Предлагается выделять следующие признаки нижнего уровня:

1. Рассматриваются вектора $\overrightarrow{AC_i},\ i = \overline{1, k+1}$. Пусть $m_i = ||\overrightarrow{AC_i}||_2,\ i = \overline{1, k+1}$. Эти вектора нормируются на свои длины соответственно:

$$\overrightarrow{c_i} = \frac{\overrightarrow{AC_i}}{m_i},\ i = \overline{1, k+1}.$$

   В качестве признаков берутся последовательно (по i) координаты получившихся векторов $\overrightarrow{c_i},\ i = \overline{1, k+1}$. Всего получается $2(k+1)$ признаков.

2. Пусть $\overrightarrow{g} = (1, 0)$ (см. рис. 14). В качестве признаков добавляются последовательно (по i) следующие ориентированные углы:

$$\angle(\overrightarrow{g}, \overrightarrow{AC_i}),\ i = \overline{1, k+1}.$$

   Получается $k+1$ признаков.

3. Также добавляются последовательно (по i) следующие ориентированные углы:

$$\angle(\overrightarrow{C_iC_{i-1}}, \overrightarrow{C_iC_{i+1}}),\ i = \overline{1, k}.$$

   Получается k признаков.

4. Наконец, в качестве признаков добавляются отношения длин соседних векторов:

$$\frac{m_i}{m_{i-1}},\ i = \overline{2, k+1}.$$

   Получается k признаков.

Таким образом, для каждого ребра e его признаковое описание f_e состоит из $5k+3$ элементов.

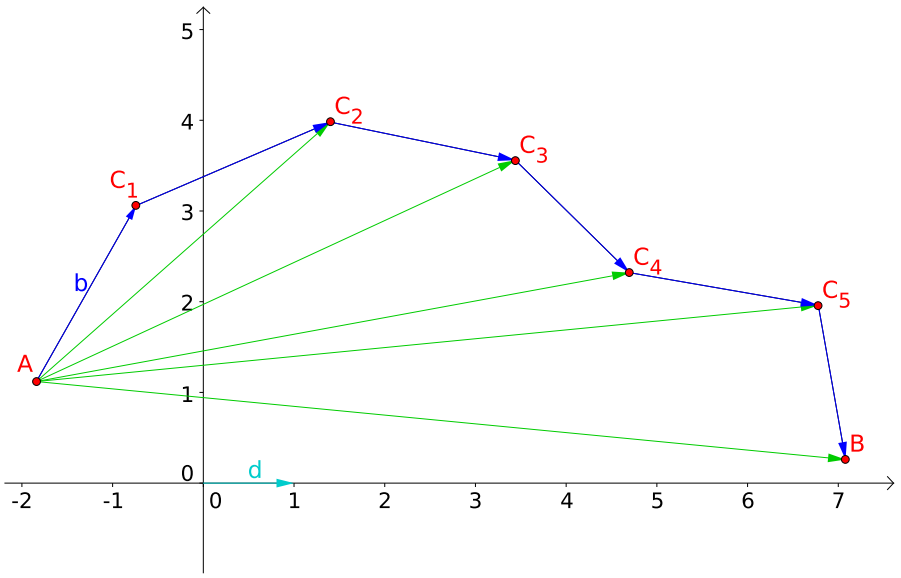


Рис. 14: Пример разбитого ребра $[A, B]$ со стартовой вершиной A

Генерация признаков для одного прямоугольника

Пусть $\mathcal{R}$ — текущая рассматриваемая прямоугольная область из разбиения рамки $\mathcal{B}$. Из соображений упорядочивания признаков вершины гиперграфа $\mathcal{S}_{agg,2}$, попавшие в $\mathcal{R}$, сортируются по полярному углу (в случае равенства полярных углов — по длине относительно нижнего левого угла $\mathcal{R}$). Обозначим признаковое описание области $\mathcal{R}$ через $f_{\mathcal{R}}$.

Будем рассматривать листья и развилки, попавшие в $\mathcal{R}$, отдельно: сначала все листья, затем все развилки. Во всех случаях стартовой вершиной будет являться текущая рассматриваемая вершина v.

1. Текущая вершина v — лист. Тогда генерируются признаки f_e для соответствующего ей ребра e, и они добавляются к итоговому признаковому описанию $f_{\mathcal{R}}$ области $\mathcal{R}$.

2. Текущая вершина v — развилка. Рассмотрим соответствующие трём выходящим рёбрам вектора $\overrightarrow{b_1}, \overrightarrow{b_2}, \overrightarrow{b_3}$. Выходящие рёбра сортируются в порядке возрастания ориентированных углов $\angle(\overrightarrow{b_i}, \overrightarrow{g}),\ i = 1, 2, 3$, после чего для них генерируюся признаки f_e. Полученные признаки добавляются в порядке сортировки рёбер к итоговому признаковому описанию $f_{\mathcal{R}}$ области $\mathcal{R}$.

Генерация признаков для графемы

Признаки нижнего уровня F_d для графемы получаются путём объединения признаков $f_{\mathcal{R}}$ в порядке упорядочивания прямоугольных областей $\mathcal{R}$.

3.2.2 Обучение классификатора

Теперь в рамках каждого признака верхнего уровня F_a имеется возможность обучить свой классификатор — каждый на своём признаковом пространстве, соответствующем своему признаковому пространству нижнего уровня F_d.

На этапе обучения берётся размеченная тренировочная выборка $(\mathbb{X}_{tr}, \mathbb{Y}_{tr})$, где $x \in \mathbb{X}_{tr}$ — бинаризованное изображение символа, $y \in \mathbb{Y}_{tr}$ — соответствующий класс изображения (буква, изображённая на нём).

Алгоритм обучения состоит из следующих шагов:

1. По всей тренировочной выборке $(\mathbb{X}_{tr}, \mathbb{Y}_{tr})$ выделить признаки верхнего уровня и по ним построить словарь классификации $\mathcal{D}$: «признак верхнего уровня $\mathcal{F}_a$ — список объектов, имеющих данный признак».

2. Для каждого уникального признака верхнего уровня $\mathcal{F}_a$ выделить те объекты, которые имеют данный признак $\mathcal{F}_a$. Для каждого из таких объектов построить признак нижнего уровня $\mathcal{F}_d$. В итоге получается новая подвыборка объектов из признакового пространства $\mathcal{F}_d$, при помощи которой обучается классификатор (см. рис 15).

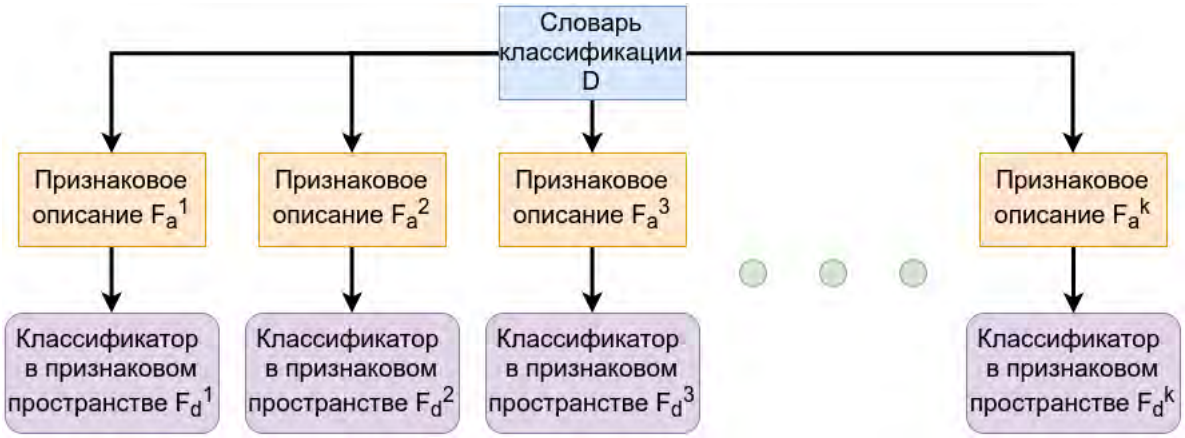


Рис. 15: Устройство словаря классификации $\mathcal{D}$

3.2.3 Алгоритм классификации

Пусть на вход приходит новый объект x (бинаризованное изображение одного символа), и его необходимо классифицировать. Для его классификации необходимо проделать следующие шаги:

1. Выделение математической модели графемы $\mathcal{S}_{agg,2}$ из x.
2. Построение признаков верхнего уровня $\mathcal{F}_a$ из $\mathcal{S}_{agg,2}$.
3. Проверка наличия признака $\mathcal{F}_a$ в словаре классификации $\mathcal{D}$, полученном на этапе обучения. Если признак не присутствует, то возвращается отказ от классификации. Если присутствует — переход к следующему шагу.
4. Построение признаков нижнего уровня $\mathcal{F}_d$, применение к нему соответствующего обученного классификатора и получение ответа.

На самом деле, шаг 3 можно значительно модифицировать. Для этого делается предположение, что признак $\mathcal{F}_a$ не присутствует в словаре классификации из-за того, что исходное изображение было не очень хорошего качества (с шумами), и агрегированный скелетный граф $\mathcal{S}_{agg,2}$ вследствие этого может содержать шумовые рёбра (см. рис. 16).

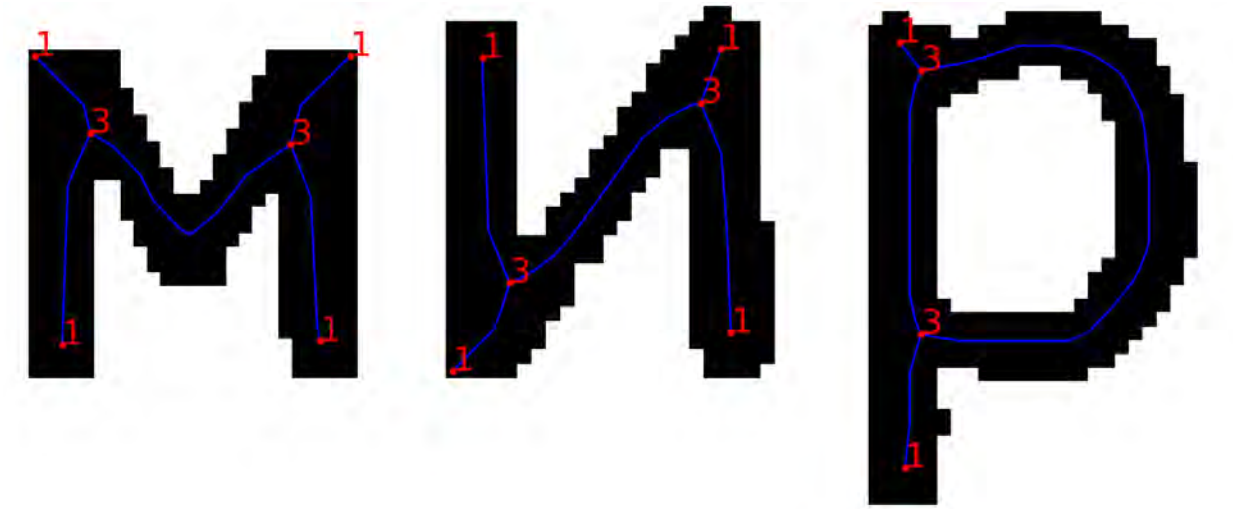


Рис. 16: Пример скелетных графов $\mathcal{S}_{agg,2}$ с шумовыми рёбрами: в первой букве два кандидата на удаление имеют длину больше порога $\mathcal{L}(\mathcal{B})$, во второй букве один из кандидатов имеет также большую длину, на третьей — кандидат на удаление всего один, поэтому по алгоритму он не отбросился.

Идея, помогающая сократить число отказов от классификации в этом случае, основывается на продолжении поиска подграфа, который возможно будет классифицировать по обученному словарю классификации $\mathcal{D}$.

Чтобы избавиться от шумовых рёбер, производятся повторные операции удаления шумовых рёбер (см. секцию 3.1.3) с поправкой на то, что теперь может быть удалено даже хотя бы одно ребро. В процессе таких действий последовательно возникают новые агрегированные скелетные графы:

$$\mathcal{S}_{agg,2},\ \mathcal{S}_{agg,3},\ \mathcal{S}_{agg,4},\ldots,\mathcal{S}_{agg,i},\ \mathcal{S}_{agg,i+1}.$$

Удаление прекращается либо когда признак верхнего уровня $\mathcal{F}_{a,i}$ для $\mathcal{S}_{agg,i}$ попал в словарь классификации (и тогда появляется возможность классифицировать графему, то есть переход к шагу 4), либо $S_{agg,i} = S_{agg,i+1}$.

В случае равенства скелетных графов можно и далее продолжать удалять шумовые рёбра, убрав из исходного алгоритма проверку на величину центрального угла аппроксимирующей дуги и увеличив порог $\mathcal{L}(\mathcal{B})$: $\mathcal{L}'(\mathcal{B}) = \kappa\mathcal{L}(\mathcal{B}),\ \kappa > 1$. Остановка процесса удаления происходит аналогично. Только после такого удаления либо осуществляется переход к шагу 4, либо возвращается отказ от классификации. Назовём такой процесс *постобработкой* $\mathcal{S}_{agg,2}$.

Итоговый алгоритм можно увидеть на рис. 17.

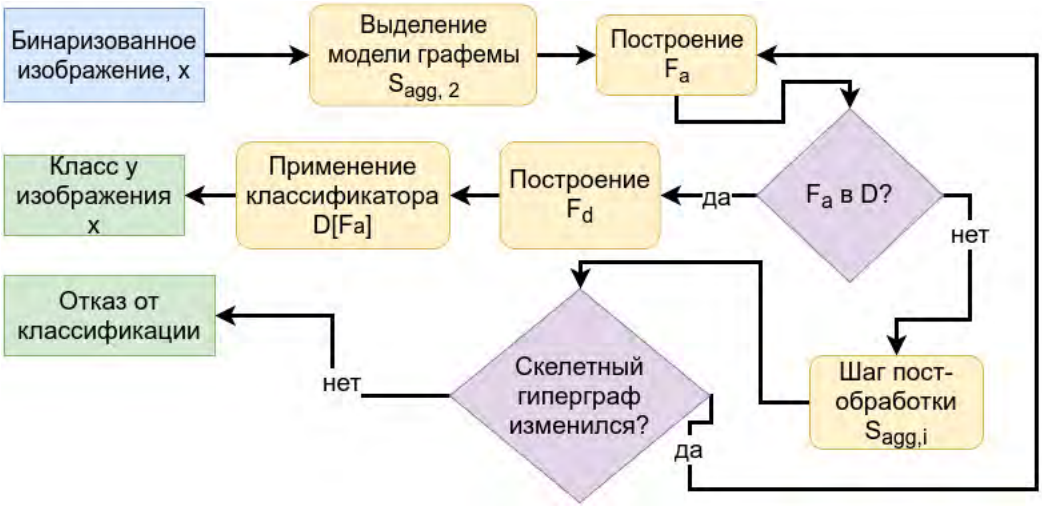


Рис. 17: Алгоритм классификации бинаризованного изображения

3.2.4 Метрика качества

В качестве метрики качества используется *точность классификации (accuracy).* Пусть a — алгоритм классификации, $(\mathbb{X}_{te}, \mathbb{Y}_{te})$ — тестовая выборка, $|\mathbb{X}_{te}| = n_{te}$, $\mathbb{X}^i_{te}$ — i-й объект тестовой выборки, $\mathbb{Y}^i_{te}$ — его истинный класс. Тогда точность классификации вычисляется по тестовой выборке соответственно следующей формуле:

$$Q(a, (\mathbb{X}_{te}, \mathbb{Y}_{te})) = \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \mathbb{I}[\mathbb{Y}^i_{te} = a(\mathbb{X}^i_{te})].$$

4 Вычислительные эксперименты

4.1 Условия экспериментов

Обучающая выборка

В качестве обучающей выборки было выбрано 88 различных шрифтов, из каждого сгенерировано по 33 буквы русского алфавита в строчном и прописном вариантах (то есть всего 66 графем) в трёх размерах шрифта: 30, 50, 100 пикселей. Генерация изображений производилась без сглаживания, то есть сразу в бинарном формате. Таким образом размер тренировочной выборки $(\mathbb{X}_{tr}, \mathbb{Y}_{tr})$ составляет $n_{tr} = 17424$ бинаризованных изображений букв. В качестве истинного класса $\mathbb{Y}^i_{tr}$ для объекта $\mathbb{X}^i_{tr}$ обучающей выборки бралась изображенная буква *в нижнем регистре.*

Приведение к нижнему регистру обосновано тем, что существует ряд букв (например, «С» и «с»), заглавное и строчное написание которых представляют собой одну графему, и предложенный алгоритм в случае прихода такой графемы теоретически

может распознать как заглавный вариант, так и строчный. Чтобы такое возможное неправильное предсказание не засчитывалось за ошибку, на этапе обучения истинные классы приводятся в нижний регистр.

Отдельно подчеркнём, что изображения в обучающей выборке могут быть совершенно разных размеров.

Свойства тестовой выборки

Пусть $(\mathbb{X}_{te}, \mathbb{Y}_{te})$ — тестовая выборка. Из соображений, приведённых выше, потребуем, чтобы $\mathbb{Y}_{te}$ состоял из букв в нижнем регистре (даже если соответствующий объект был изображён в верхнем). Так же, как и в обучающей выборке, изображения из $\mathbb{X}_{te}$ могут быть разных размеров.

Параметры предложенного алгоритма

Ниже приведены параметры, которые использовались на стадиях обучения и предсказания для предложенного алгоритма:

1. Параметр стрижки $\alpha = 0.06 \cdot \mathcal{B}_H$.

2. Порог для отсечения по длине на стадии удаления шумовых рёбер:
$$\mathcal{L}(\mathcal{B}) = \frac{2}{7}\max(\mathcal{B}_H, \mathcal{B}_W).$$

3. Порог для отсечения по длине на стадии постобработки увеличивается в 1.8 раза:
$$\mathcal{L}'(\mathcal{B}) = 1.8 \cdot \mathcal{L}(\mathcal{B}) \quad (\kappa = 1.8).$$

4. Порог для отсечения по центральному углу на стадии удаления шумовых рёбер:
$$\mathcal{A} = \frac{\pi}{5}.$$

5. На стадии извлечения признаков верхнего уровня полагается $n = m = 3$.

6. На стадии извлечения признаков нижнего уровня фиксируемая сетка полагается равной:
$$\mathcal{Z}_8 = \left[\frac{1}{50}, \frac{1}{5}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{4}{5}\right].$$

7. В качестве классификаторов нижних уровней рассматривается *случайный лес (Random forest)* [10].

Базовый алгоритм

В качестве базового алгоритма была выбрана свёрточная нейронная сеть [11][12], архитектура которой представлена на рис. 18:

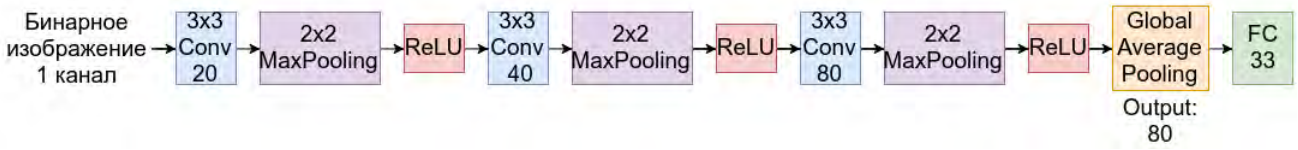


Рис. 18: Архитектура нейронной сети

Расшифровка обозначений:

- $k \times k$ Conv f — свёрточный слой с ядром размера $k \times k$ и f выходными фильтрами (каналами);
- $k \times k$ MaxPooling — max-пулинг слой с ядром размера $k \times k$;
- ReLU — ReLU нелинейность;
- Global Average Pooling — global average-пулинг слой;
- FC (Fully Connected) m — полносвязный слой с выходным слоем из m нейронов [12].

Из соображений решения задачи классификации над выходным слоем $(x^1, x^2, \ldots, x^{33})$ из 33 нейронов производится softmax-активация:

$$y^j = \text{softmax}(x^j) = \frac{e^{x^j}}{\sum\limits_{j=1}^{33} e^{x^j}}, \quad j = \overline{1, 33}.$$

Пусть C — количество классов в задаче классификации.

В качестве оптимизируемой функции потерь берётся кросс-энтропия:

$$\mathcal{L}(y_i, \hat{y}_i) = -\sum_{j=1}^{C} y_i^j \log \hat{y}_i^j,$$

$$\mathcal{L}(y, \mathbb{Y}_{tr}) = \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \mathcal{L}(y_i, \hat{y}_i),$$

где $y_i \in [0, 1]^C$ — предсказание сети на i-м объекте, y — предсказание сети на всей тренировочной выборке, $\hat{y}_i$ — вектор, описывающий наблюдаемое значение: $\hat{y}_i \in [0, 1]^C$, $\sum_{j=1}^{C} \hat{y}_i^j = 1$ и если i-й объект имеет класс j (т.е $\mathbb{Y}_{te}^i = j$), то $\hat{y}_i^j = 1$.

Замечание. Так как входные изображения могут быть разных размеров, на этапе обучения данные в нейросеть подавались батчем, состоящим из 1 изображения.

Обучение сети происходило со следующими параметрами:

- число эпох = 200;
- используемый метод оптимизации для настройки весов нейронной сети — adaptive moment estimation (Adam) [13];
- сохранялась та конфигурация весов, на которой нейронная сеть показывала наилучший результат на валидационной выборке.

Используемое программное обеспечение

Построение скелетного графа бинаризованного изображения реализовано на языке C++. Построение математической модели графемы и предлагаемый в работе алгоритм классификации был реализован с помощью средств языка Python3. Выбранная в качестве базового алгоритма свёрточная нейронная сеть была реализована посредством модуля PyTorch [14] языка Python3.

4.2 Эксперименты

Эксперимент 1

В этом эксперименте в качестве тестовой выборки берутся те же самые 88 шрифтов, которые использовались при обучении, но другого размера шрифта, который равен 80 пикселям. Таким образом, $n_{te} = 5800$. Изображения букв генерируются при помощи программы, то есть изображения качественные, без шумов и бинаризованные. Результаты работы двух методов (структурный анализ — описываемый в работе метод распознавания) представлены в таблице 1:

	Структурный анализ	**Нейронная сеть**
Качество, Q	0.99689	**0.99862**
Доля отказов от классификации	0.00086	0

Таблица 1: Результаты работы двух методов

Эксперимент 2

В этом эксперименте в качестве тестовой выборки берутся 50 шрифтов, которые *не использовались* при обучении [15]. Размер шрифта равен 80 пикселям. Размер тестовой выборки $n_{te} = 3300$. Изображения букв генерируются при помощи программы, то есть изображения качественные, без шумов и бинаризованные. Результаты работы двух методов представлены в таблице 2:

	Структурный анализ	**Нейронная сеть**
Качество, Q	**0.97**	0.96515
Доля отказов от классификации	0.01364	0

Таблица 2: Результаты работы двух методов

Эксперимент 3

В этом эксперименте в качестве тестовой выборки берутся те же самые 50 шрифтов, что и в предыдущем эксперименте 4.2, и того же размера. Сначала генерируется документ (.doc) со всеми буквами из тестовой выборки, затем этот документ конвертируется в изображение в формате png с разрешением 300 dpi. То есть изображения получаются довольно качественные, без посторонних шумов, но буквы сглажены и небинаризованы (цветные) (см.рис. 19 и 20). Сначала изображения из цветного представления RGB переводились в серые тона Y по формуле:

$$Y = 0.299R + 0.587G + 0.114B.$$

Затем изображения были бинаризованы при помощи метода Оцу [16].

Рис. 19: Пример входного изображения

Рис. 20: Пример буквы из исходного изображения

Результаты работы двух методовв представлены в таблице 3:

	Структурный анализ	**Нейронная сеть**
Качество, Q	**0.94818**	0.94454
Доля отказов от классификации	0.01485	0

Таблица 3: Результаты работы двух методов

Эксперимент 4

В этом эксперименте в качестве тестовой выборки берутся 18 выборочных шрифтов из 50 шрифтов эксперимента 4.2. Размер шрифтов полагается равным 80 пикселям. Размер тестовой выборки $n_{te} = 1188$. Сначала генерируется документ (.doc) со всеми буквами из тестовой выборки, затем этот документ распечатывается на принтере. Далее полученные образцы сканируются с разрешением 300 dpi. То есть изображения получаются менее качественные, нежели чем в предыдущем случае: появляются шумы из-за вывода текста на бумагу, а также при последующем сканировании (см.рис. 21 и 22). Заметим также, что изображения являются цветными. Как и в предыдущем эксперименте, изображения переводились в серые тона, а затем бинаризовывались при помощи метода Оцу.

Рис. 21: Пример входного изображения

Рис. 22: Пример буквы из исходного изображения

Результаты работы двух методов представлены в таблице 4:

	Структурный анализ	**Нейронная сеть**
Качество, Q	**0.95538**	0.94696
Доля отказов от классификации	0.01263	0

Таблица 4: Результаты работы двух методов

5 Анализ экспериментов

Проведённые эксперименты показывают, что:

- по качеству предлагаемый метод не уступает выбранному базовому алгоритму;
- в случае, когда на вход подаются шрифты не из тренировочной выборки, предлагаемый метод показывает результаты лучше, чем базовый алгоритм;
- чем менее качественные изображения подаются на вход, тем качество распознавания обоими алгоритмами в среднем ниже;
- предлагаемый метод имеет небольшую долю отказов от классификации, которая растёт с ухудшением качества изображения.

6 Выводы

Проведённые эксперименты подтверждают, что предлагаемая математическая модель графемы показала свою работоспособность. Более точно, из неё можно выделить признаки, достаточные для качественного распознавания изображений букв

в большинстве шрифтах. Также это означает, что выполнены свойства графемы, то есть формализацию понятия графемы можно считать успешной.

Достоинства предлагаемого метода распознавания букв:

1. Независимость от размера, типа шрифта и типа начертания буквы.
2. Выделение общей структуры (математической модели графемы) букв, которой достаточно для распознавания букв в новых шрифтах.
3. Интерпретируемость признаков.

Недостатки метода:

1. Наличие отказов от классификации.
2. Зависимость качества распознавания от качества бинаризации изображения.

7 Заключение

В данной работе предложена формализация понятия «графема», а именно предложена математическая модель графемы, удовлетворяющая свойствам графемы. На основе этой модели предложен метод генерации признаков, использующихся для последующего построения алгоритма классификации изображений букв (то есть определена мера сходства между математическими моделями графем). По этому походу к извлечению признаков был сделан доклад на международной научной конференции студентов, аспирантов и молодых учёных «Ломоносов-2018» [17]. Также в данной работе предложен алгоритм распознавания текста на изображении.

Предложенный алгоритм показывает высокие результаты распознавания, проведённые эксперименты это подтверждают.

Задачами дальнейшего исследования являются:

- Улучшение признаков верхнего и нижнего уровней.
- Решение проблемы наличия отказов от классификации.
- Модификация итерационной части (постобработки) алгоритма классификации.

Список литературы

[1] *ParaType*. Цифровые шрифты. — ParaType. — 2008.

[2] *О. В. Осетрова*. Семиотика шрифта. — Вестник Воронежского государственного университета. Серия: Филология. Журналистика. — 2006.

[3] *А. Соломоник*. О языке и языках. — М.: Издательство «Спутник+». — 2017.

[4] *А. А. Зализняк*. Русское именное словоизменение приложением избранных работ по современному русскому языку и общему языкознанию. — М.: Языки славянской культуры. — 2002.

[5] *Ю. Гордон*. Книга про буквы от аа до яя. — М. Изд-во Студии Артемия Лебедева. — 2017.

[6] Страница с примерами распознавания графем компанией abbyy. — `https://habr.com/company/abbyy/blog/228251/`.

[7] Страница с примерами методов распознавания текста. — `https://habr.com/post/220077/`.

[8] *Л.М. Местецкий*. Непрерывная морфология бинарных изображений екста й abbyy по современному . — М.: ФИЗМАТЛИТ. — 2009.

[9] *Felix Hausdorff*. Grundzüge der mengenlehre. — 1914.

[10] Kam, H. T. Random decision forests / Ho Tin Kam // Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC. — 1995.

[11] Gradient-based learning applied to document recognition / LeCun Yann, Léon Bottou, Yoshua Bengio, Patrick Haffner // Proceedings of the IEEE. — 1998.

[12] *Bishop C. M.* Pattern recognition and machine learning. — Springer. — 2006.

[13] Kingma, D. P. Adam: A Method for Stochastic Optimization / Diederik P. Kingma, Jimmy Ba // arXiv preprint arXiv:1412.6980. — 2014. — `https://arxiv.org/abs/1412.6980`.

[14] Pytorch. — `https://pytorch.org/`.

[15] Fonts database. — `https://www.fontsquirrel.com/`.

[16] Otsu, N. A threshold selection method from gray-level histograms / N. Otsu // IEEE Trans. Sys., Man., Cyber. — 1979.

[17] *А. Л. Липкина.* Генерация структурных признаков для классификации букв в цифровых изображениях / *А. Л. Липкина* // Сборник тезисов XXV Международной научной конференции студентов, аспирантов и молодых ученых «ЛОМОНОСОВ-2018». — Макс-Пресс, 2018.